

Figure 2. Data Encapsulation Through the Network Stack (Prior Art)

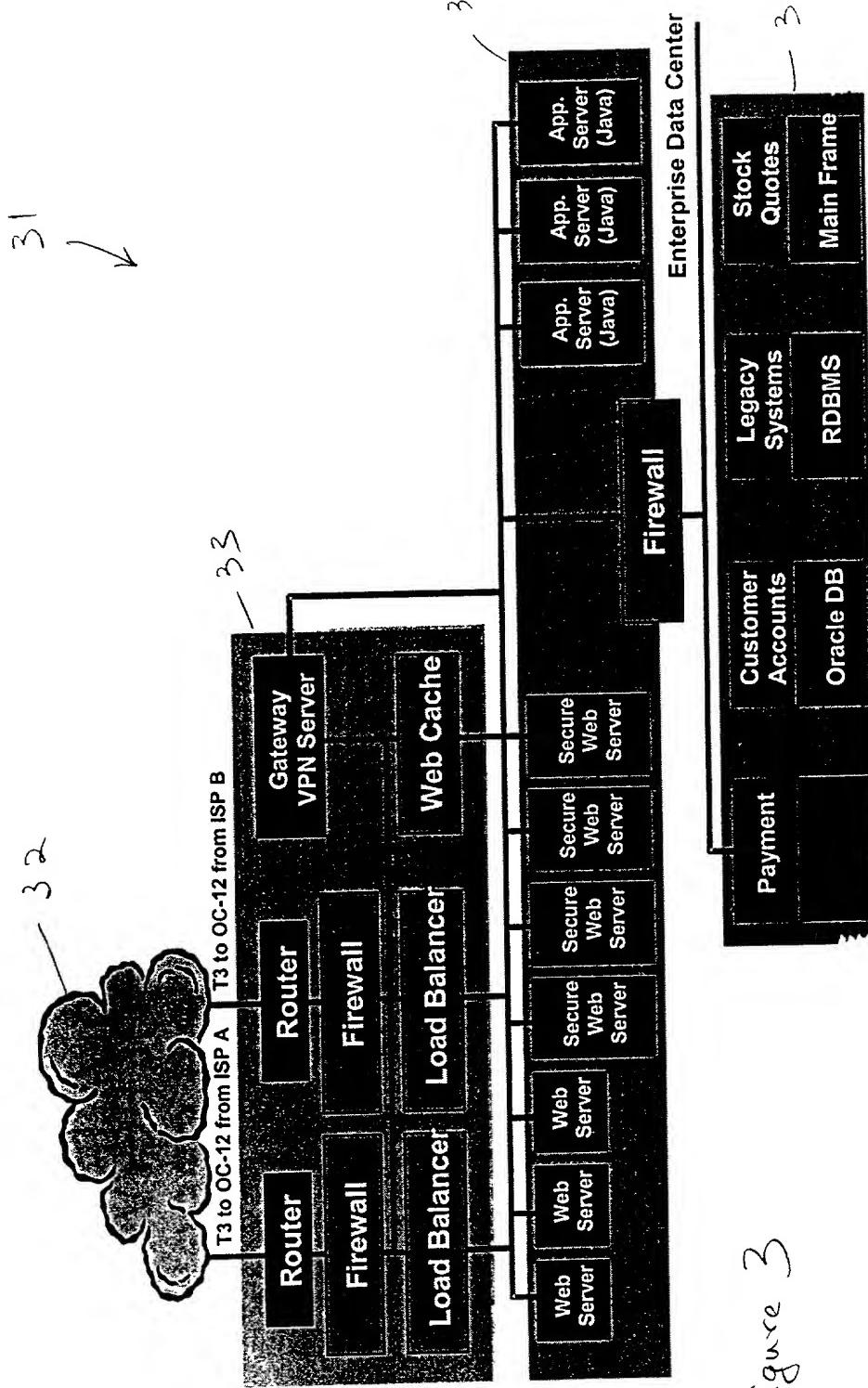


Figure 3

Fig. 4

(Prior  
Art)

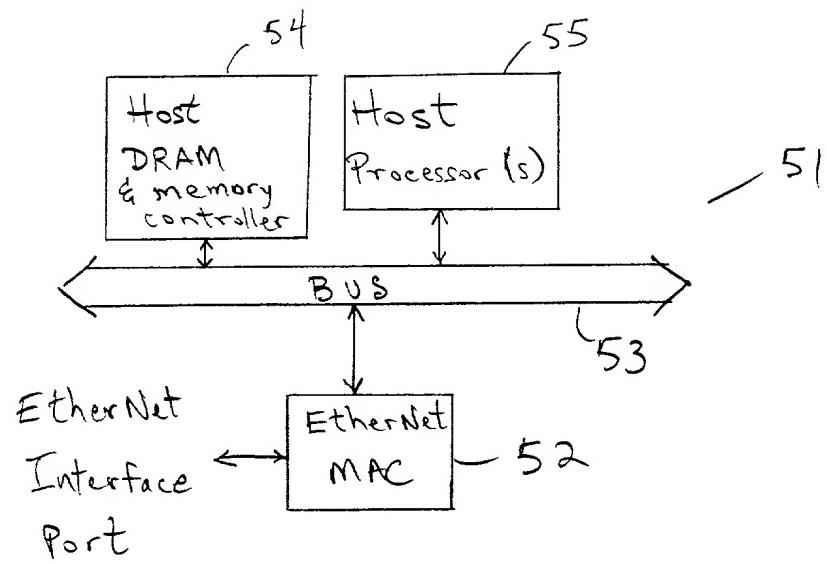
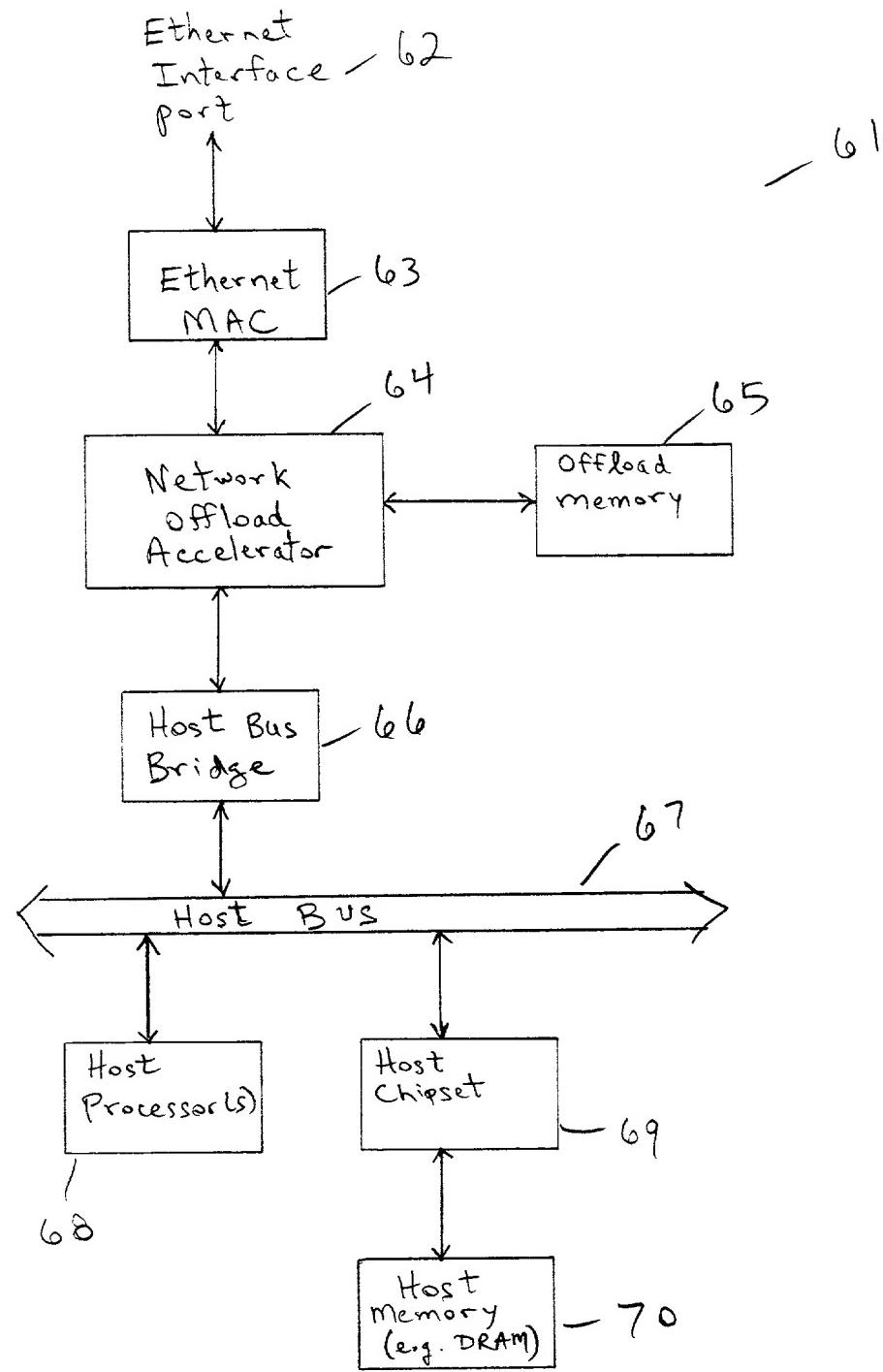


Figure 5 (prior art)



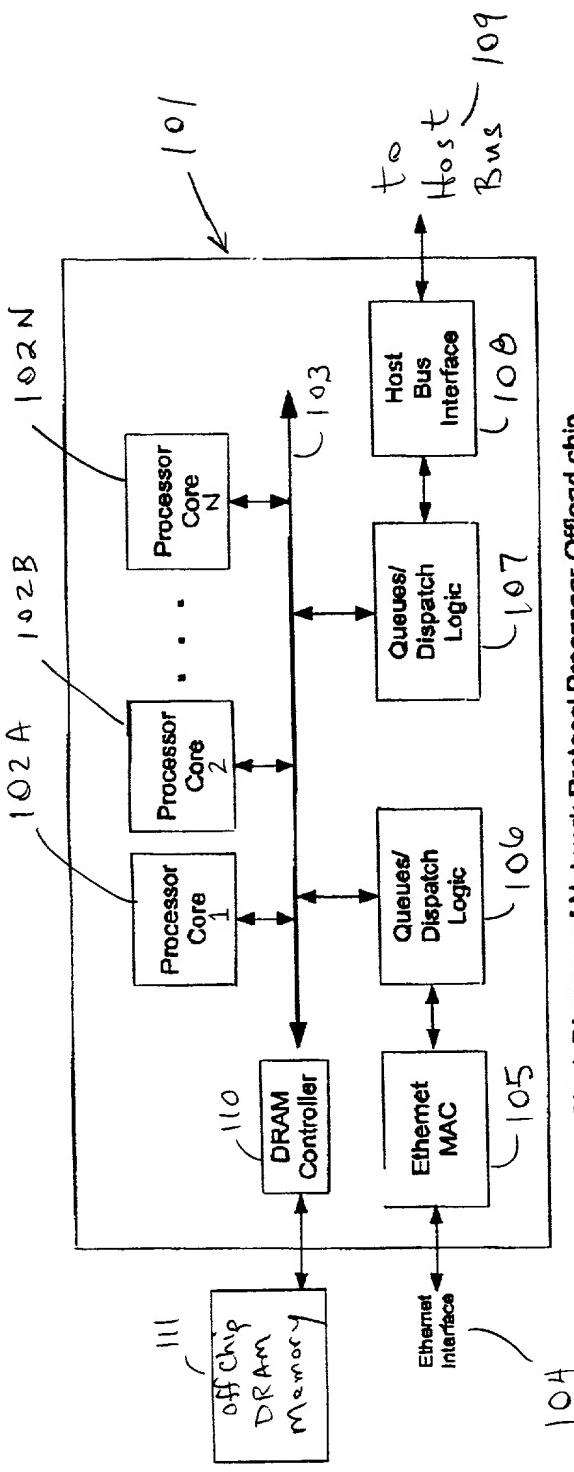
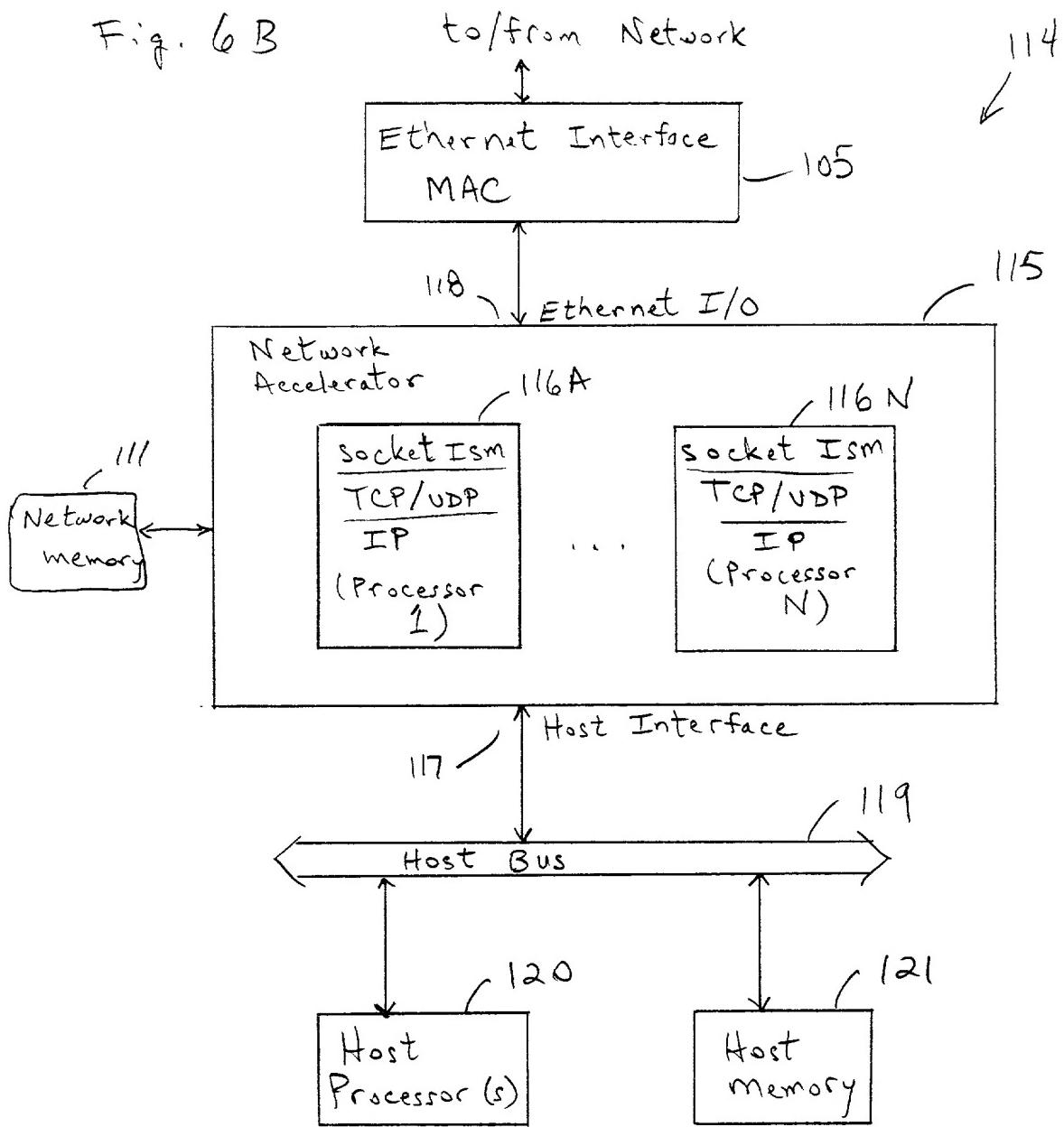


Fig. 6A Block Diagram of Network Protocol Processor Offload chip

Fig. 6 B



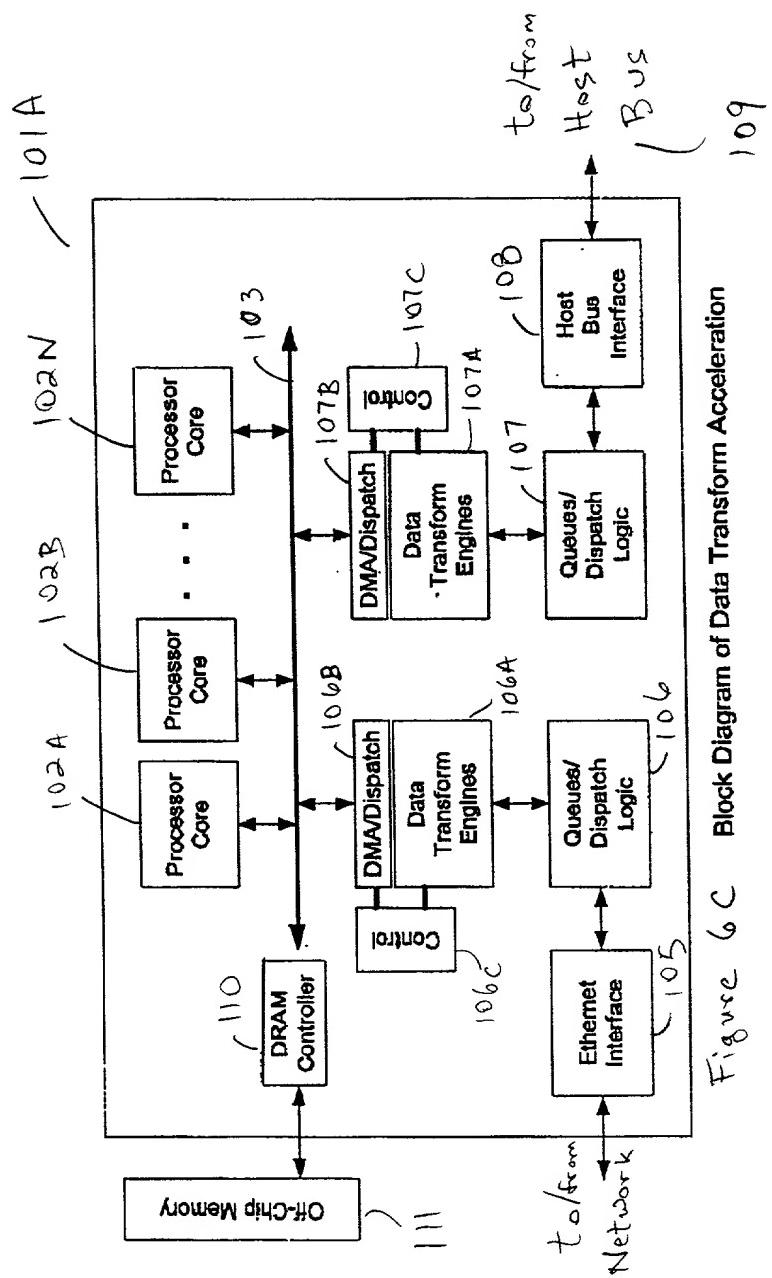


Figure 6C Block Diagram of Data Transform Acceleration

109

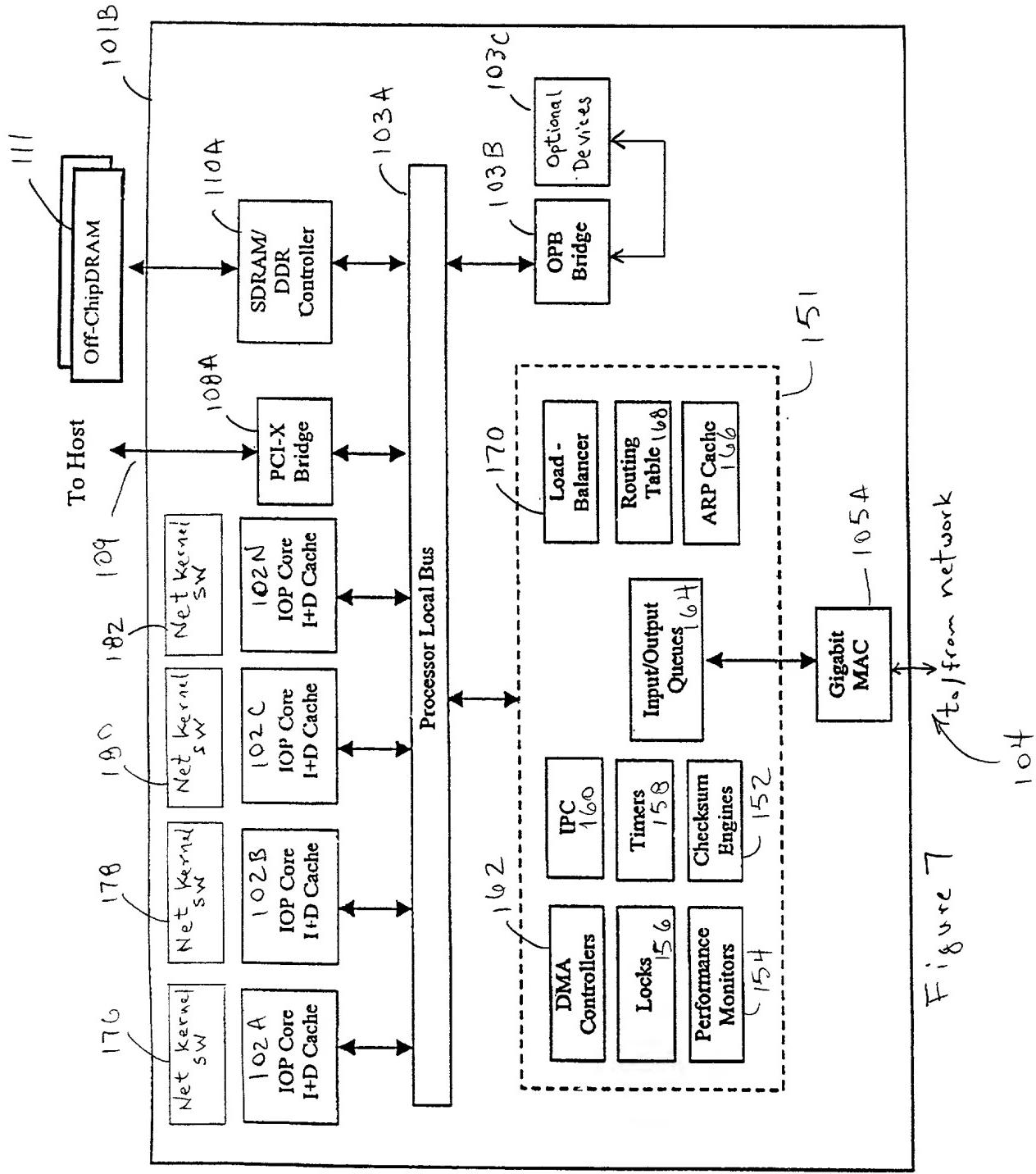


Figure 7 *Architecture from network*

Fig. 8A

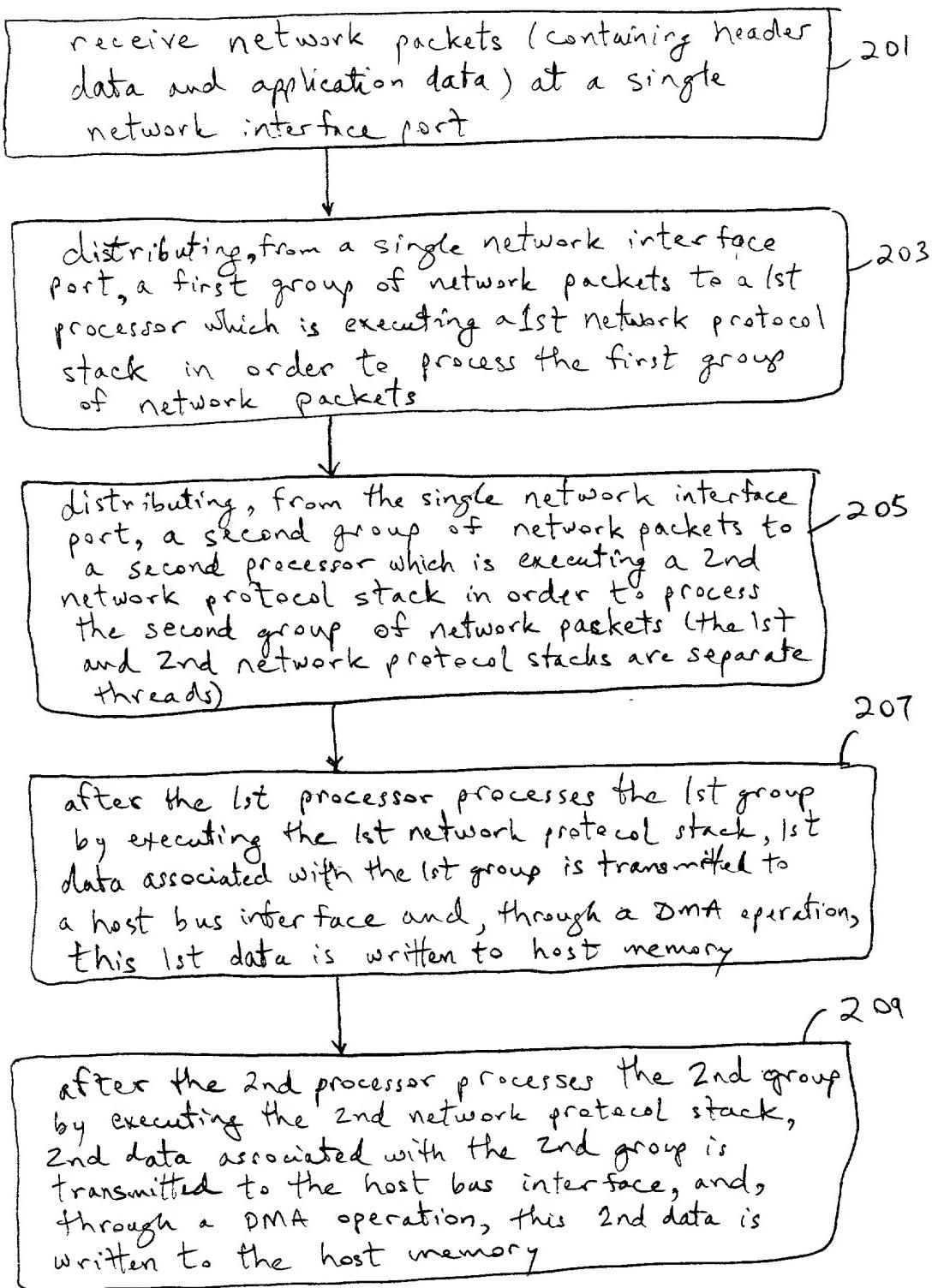


Fig. 8.3

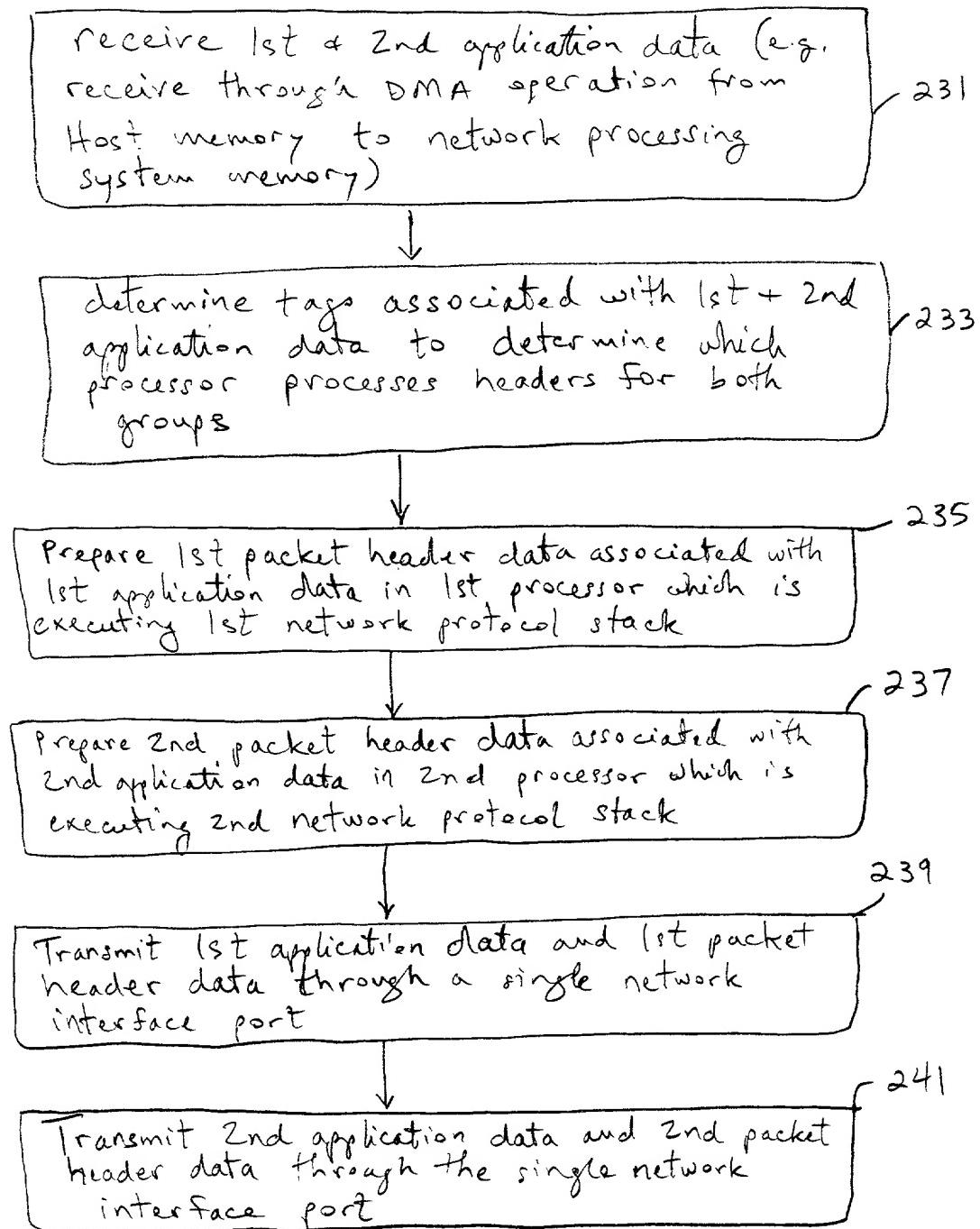


Fig. 8c

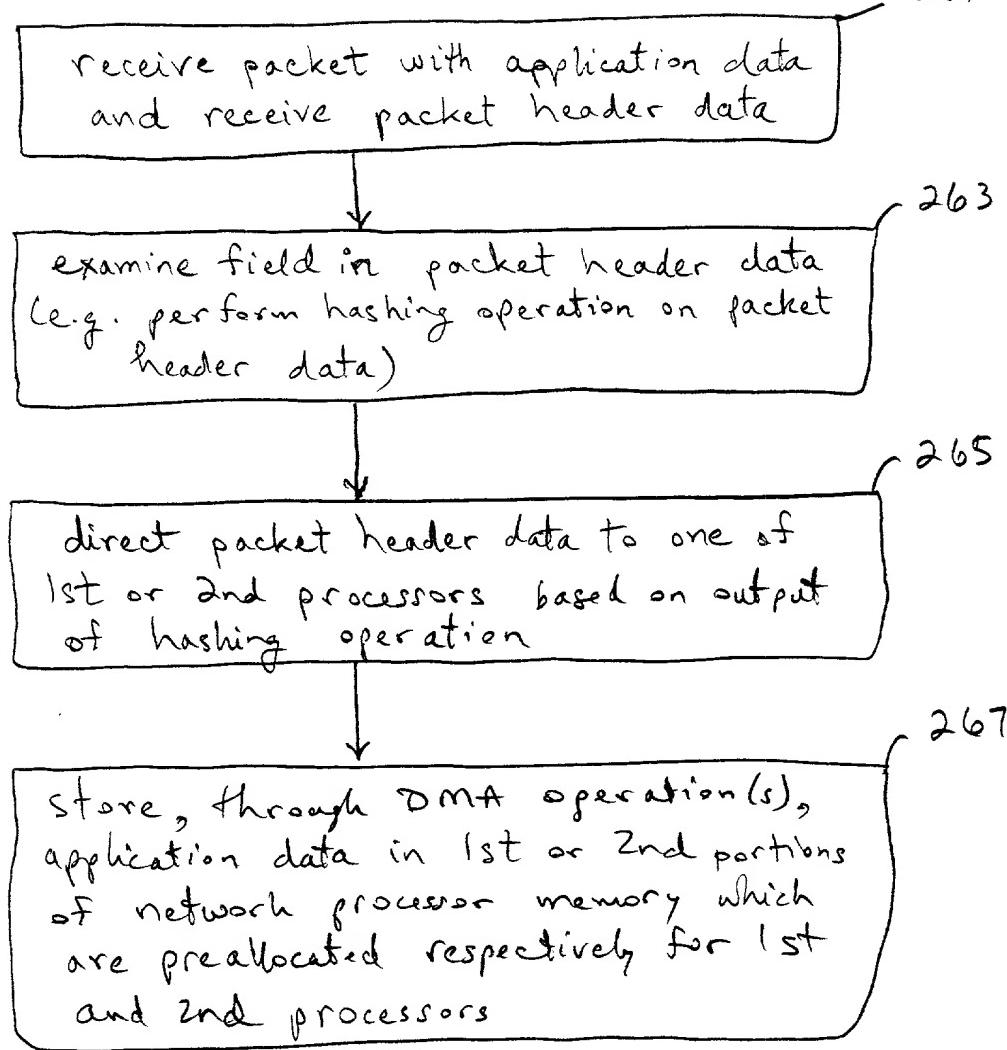


Fig. 9A

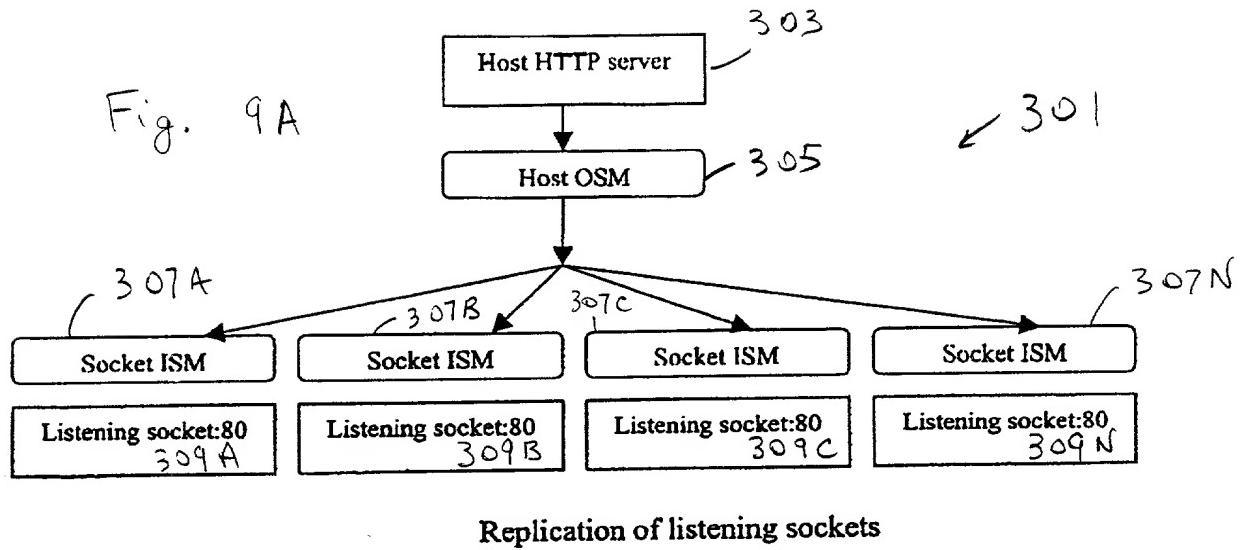


Fig. 9B

312

Initialize OSM and ISM

(e.g. ISMs transmit number of network protocol processors (IOPs) and each ISM transmits an IOP-specific handle to the OSM)

314

After initialization, a host application establishes a socket connection, using socket API calls

316

A client application connects to the socket the server application is listening on

318

The OSM then receives the client request, invoking the proper server application by identifying it from the OSM handle it allocated previously, and recording the associated ISM handle and IOP number in a table. The server application builds a response to the client. It transmits this request to the client, via the proper IOP, by prefixing the response using an { OSM handle, ISM handle } message directed to the proper IOP by posting it to the IOP's IQP.

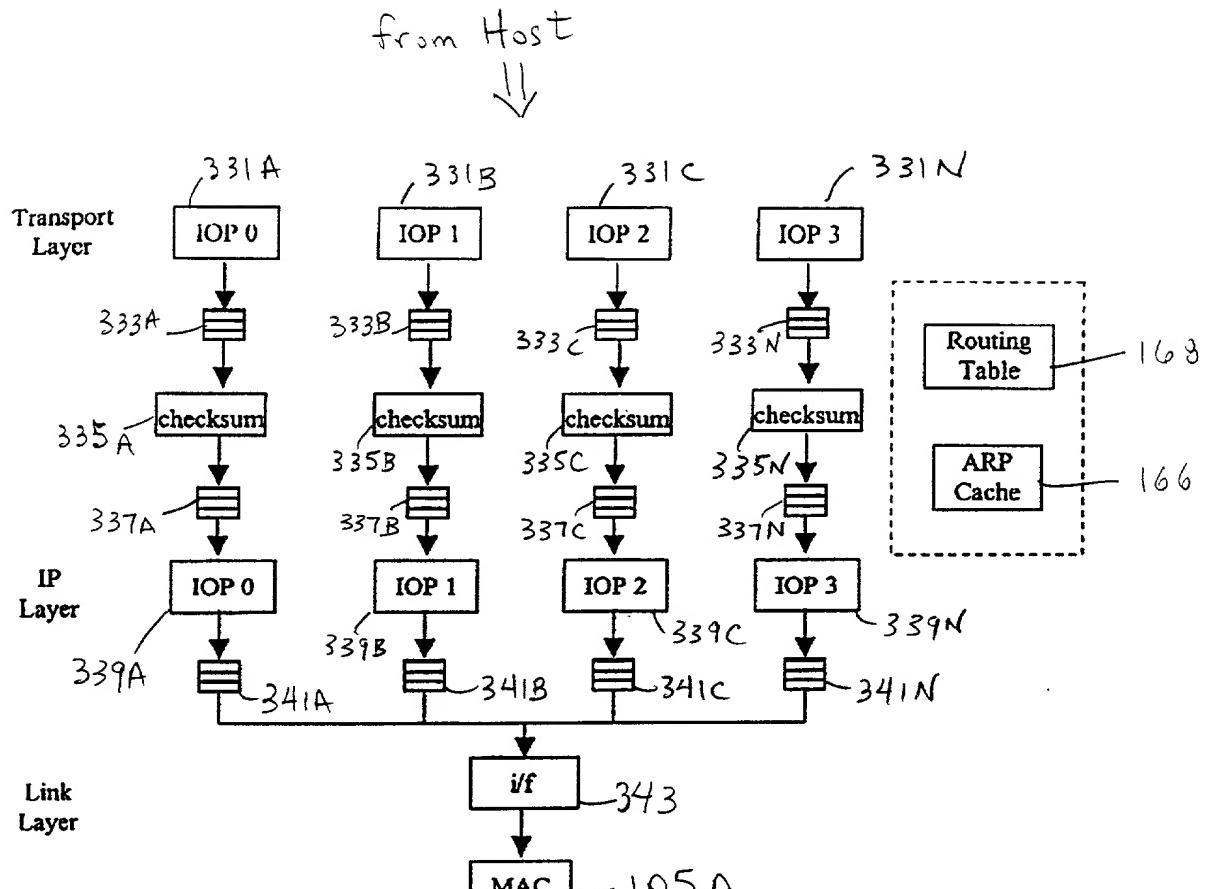


Figure 10A

The packet-sending path.

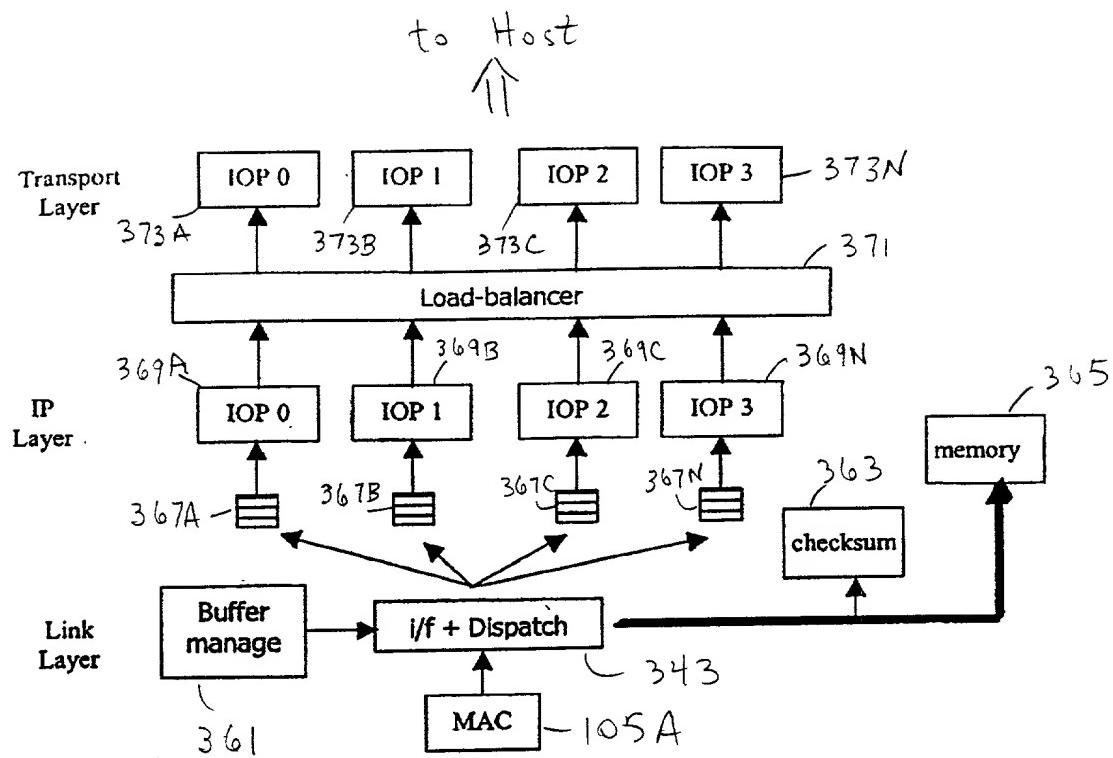


Figure 10B  
The packet-receiving path.

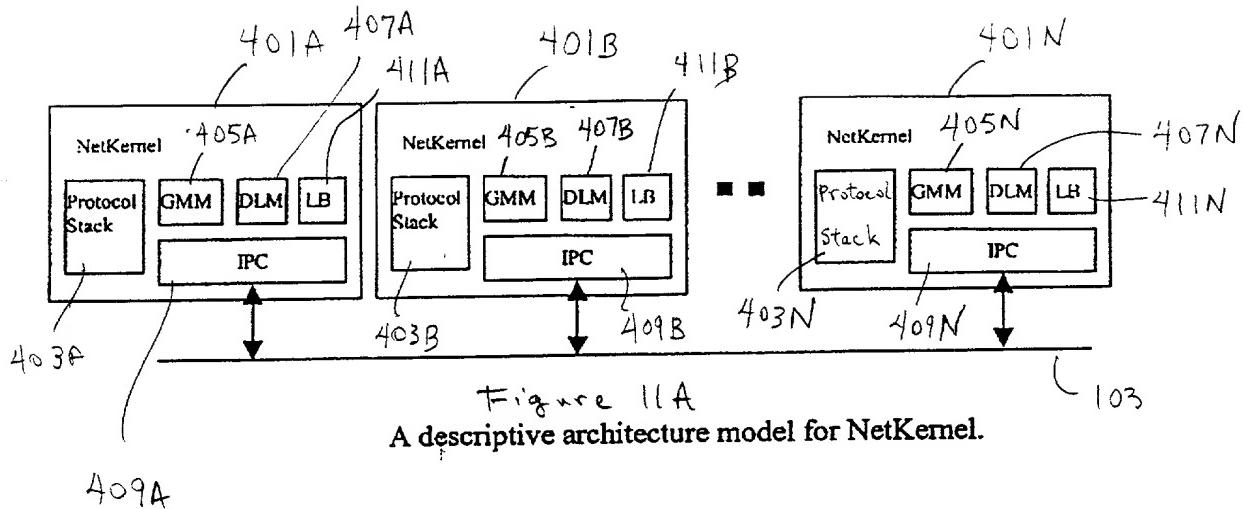
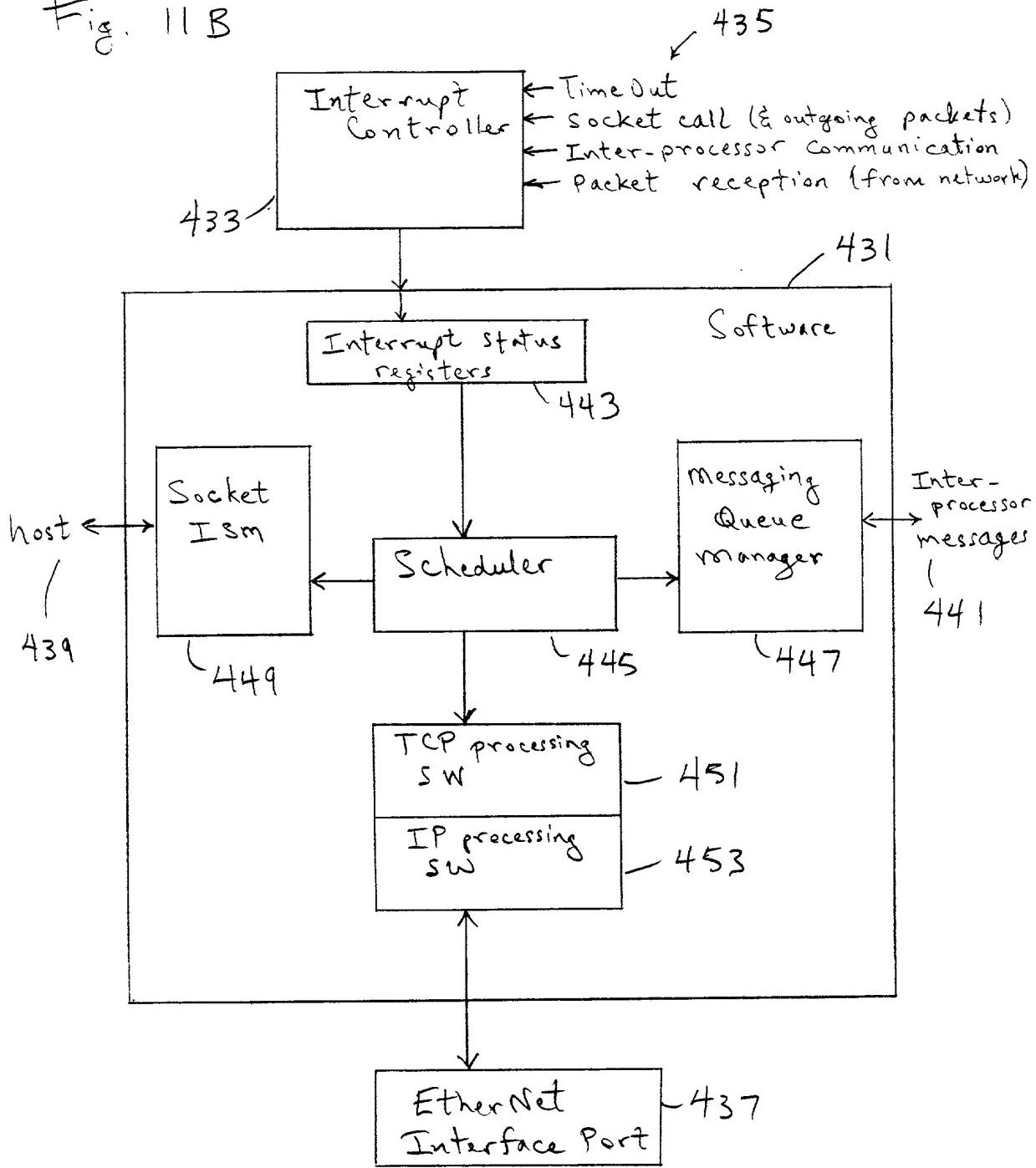


Figure 11A  
A descriptive architecture model for NetKernel.

Fig. 11B



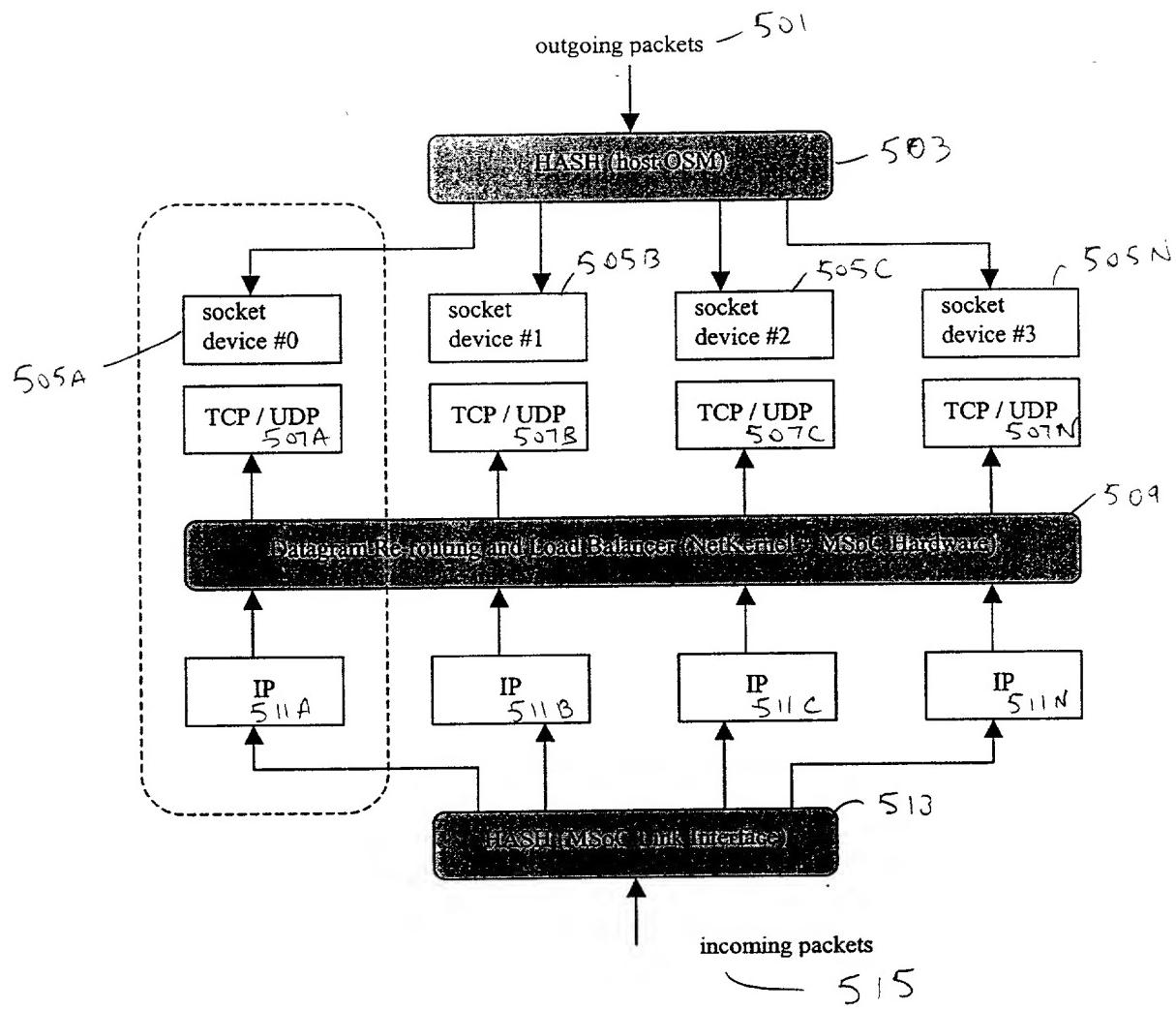


Figure 12A

An illustration of packet flows for load balancing.

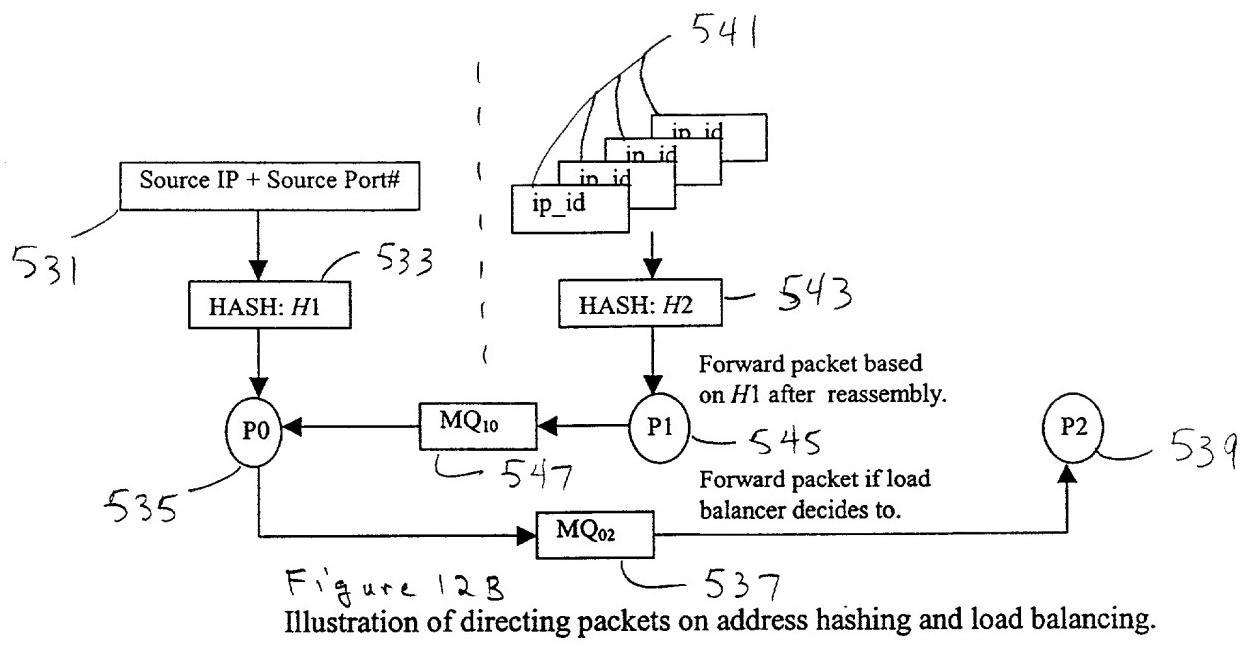


Figure 13

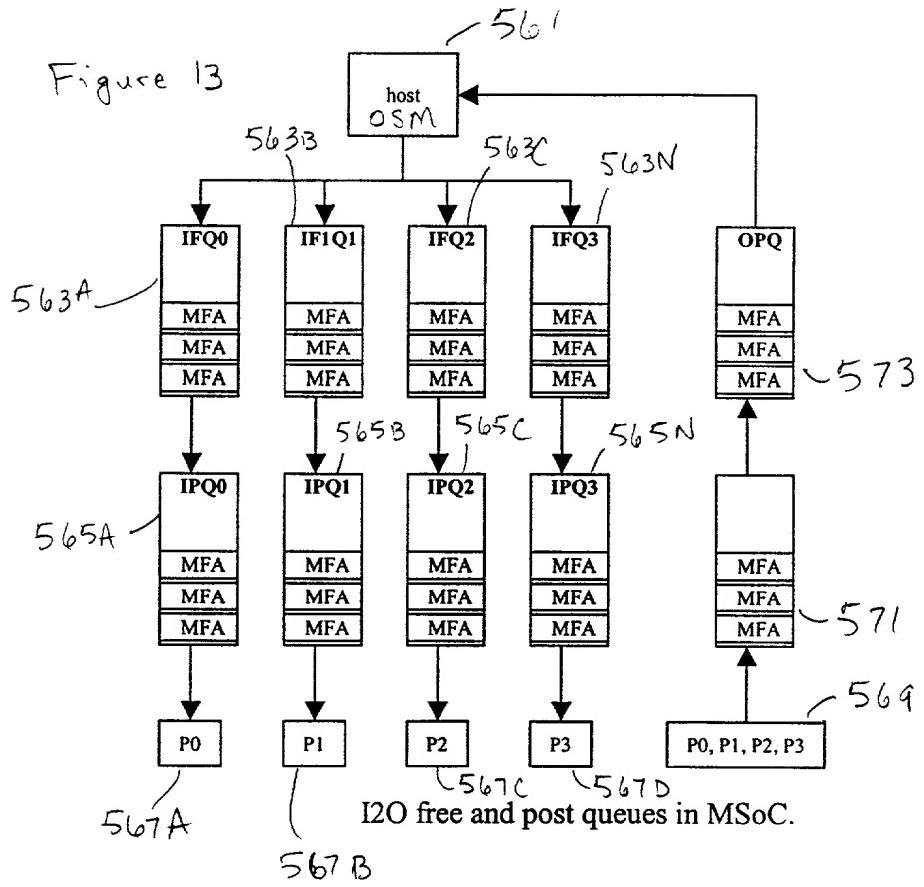


Figure 14

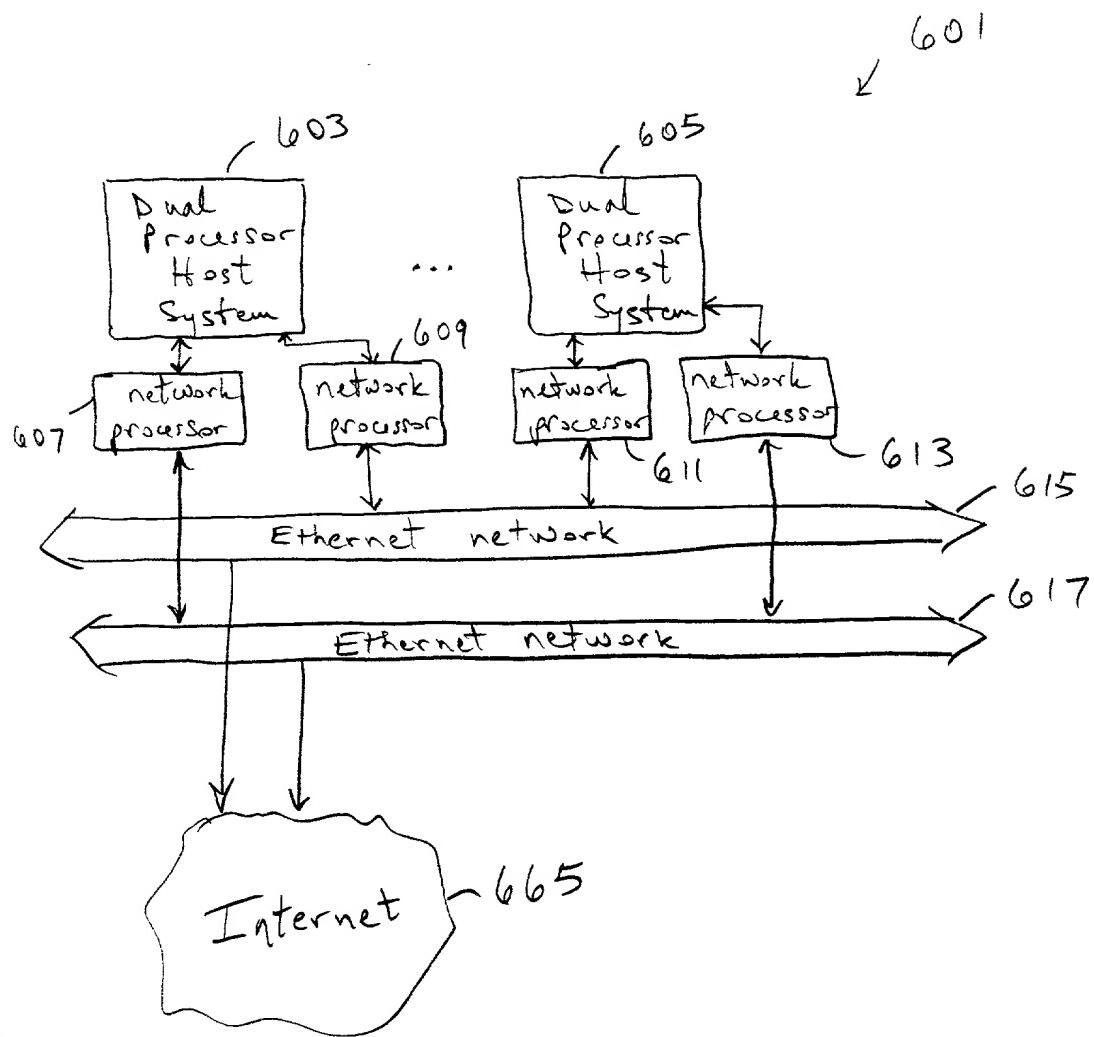
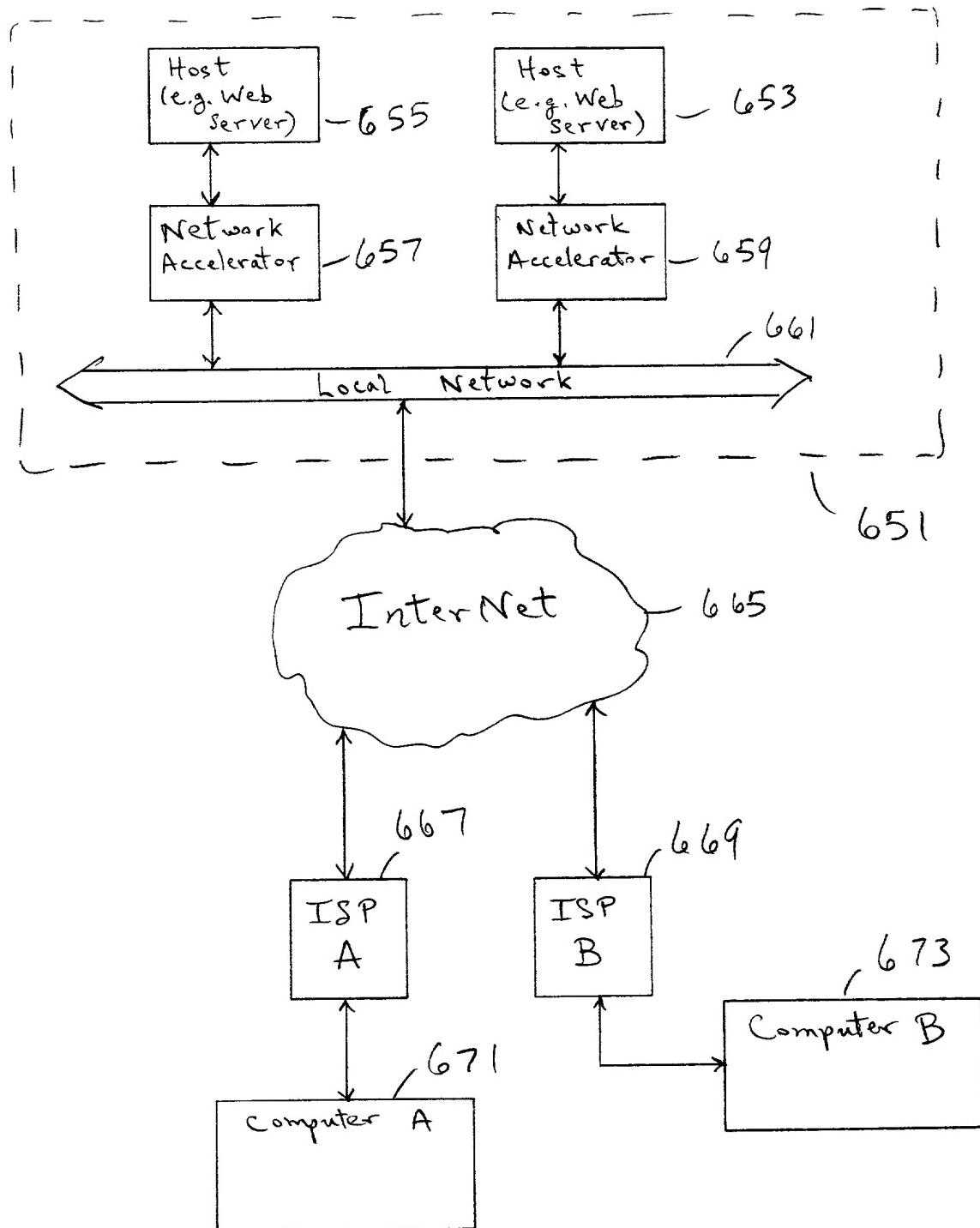


Fig. 15



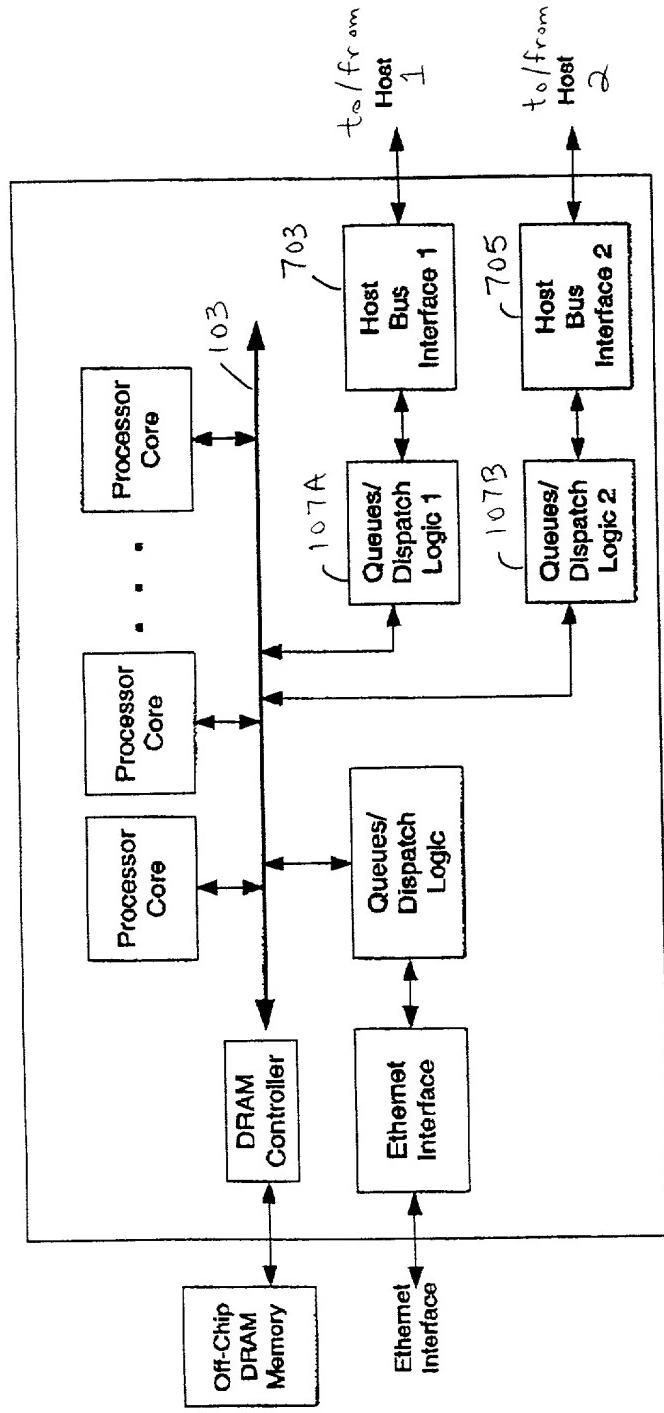
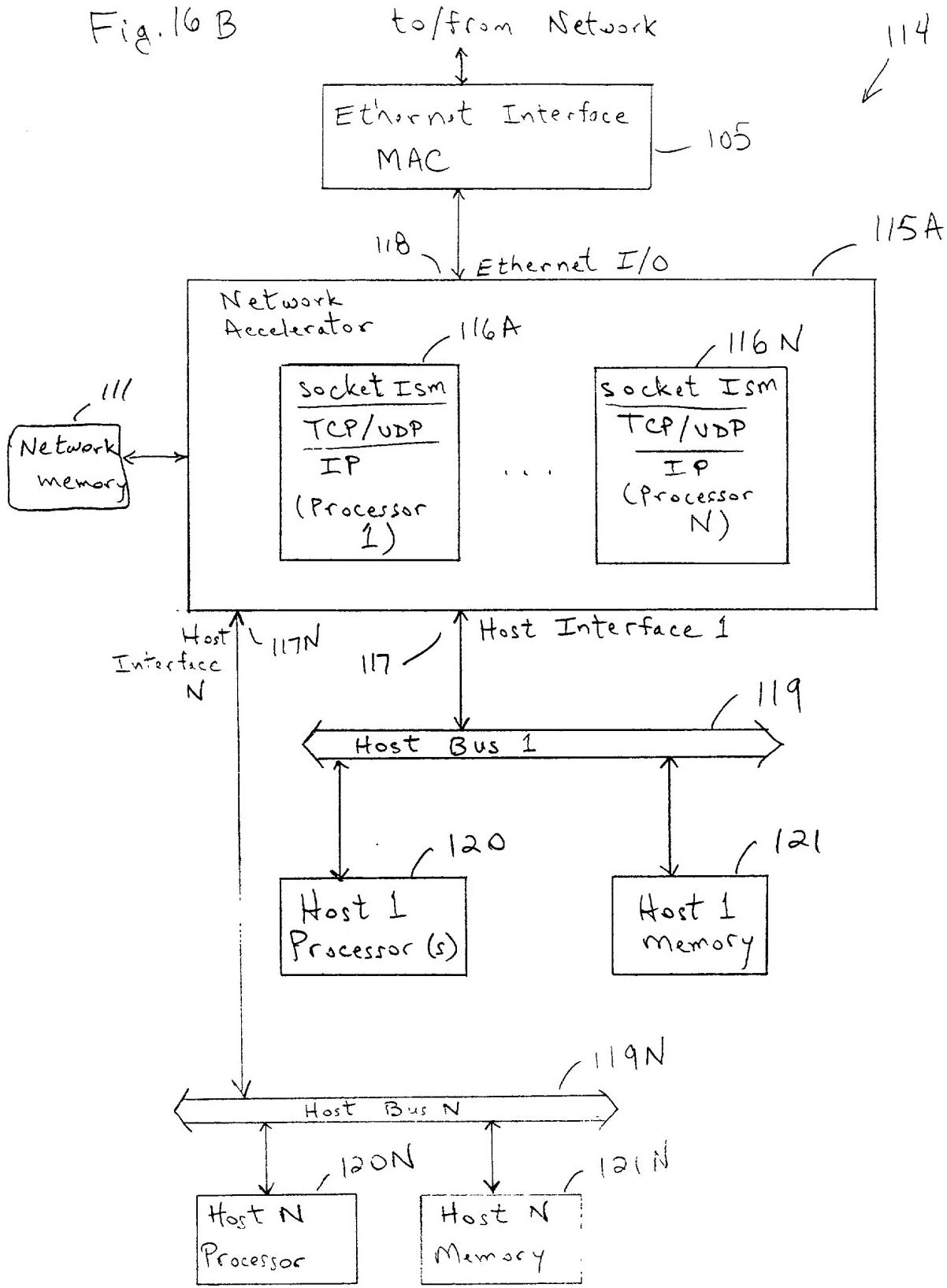


Fig. 16 A

Fig. 16 B



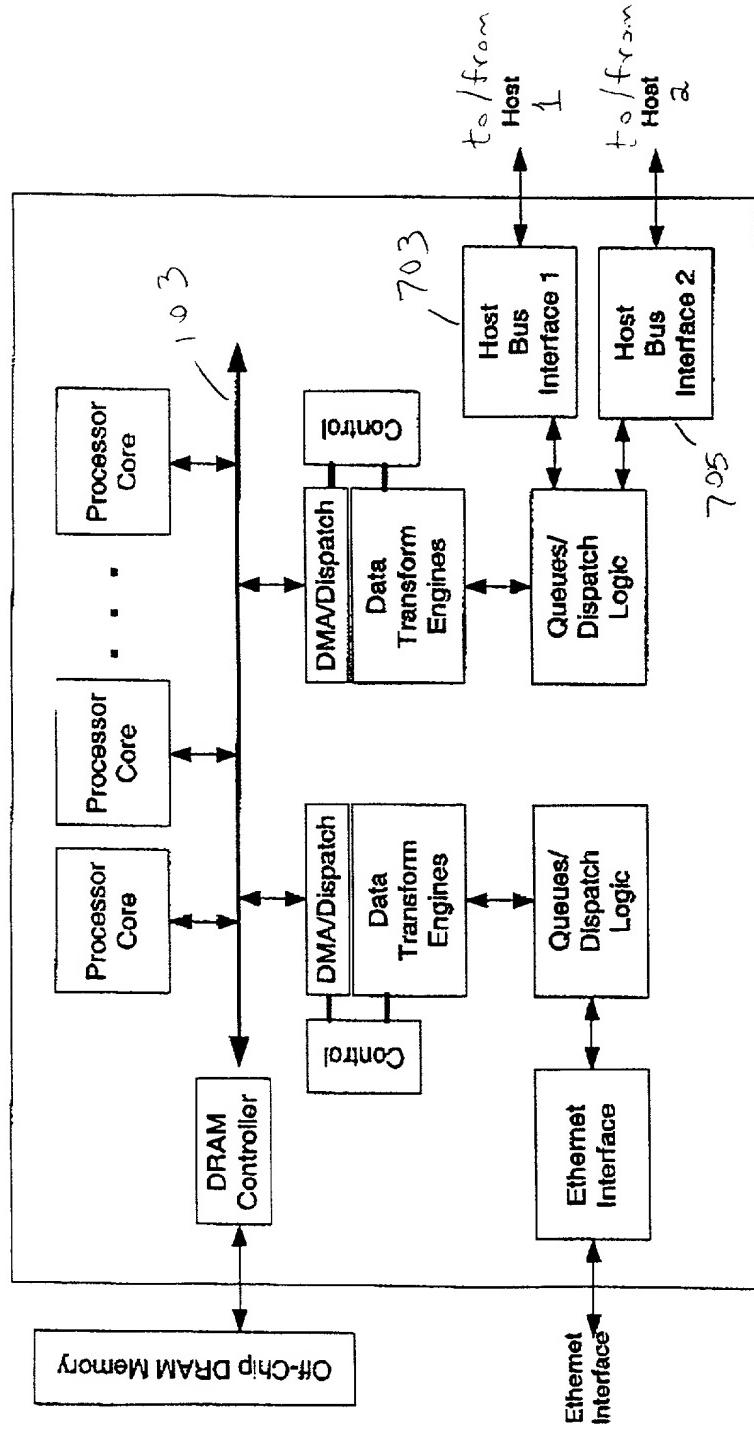
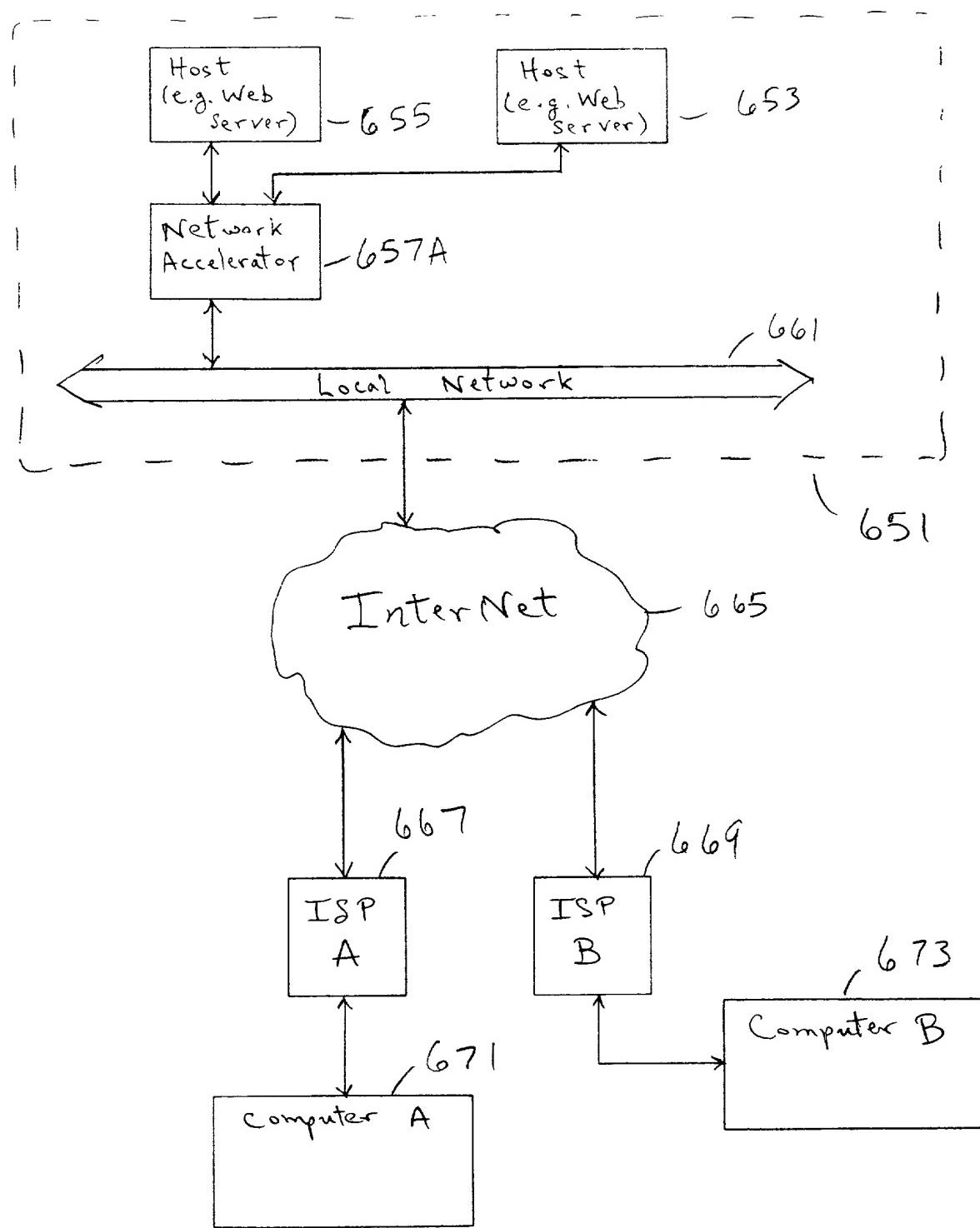


Fig. 16C 711

Fig. 16D



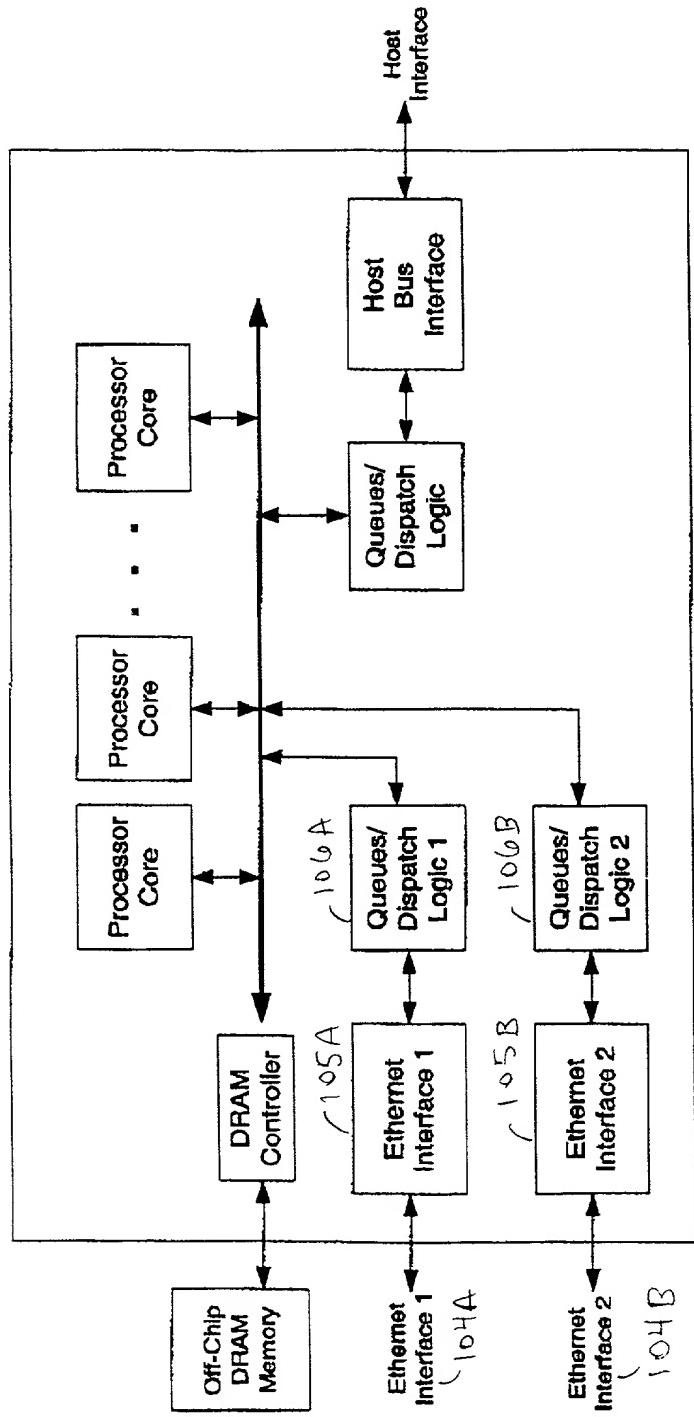


Fig. 17 A  
751

Figure 17 B

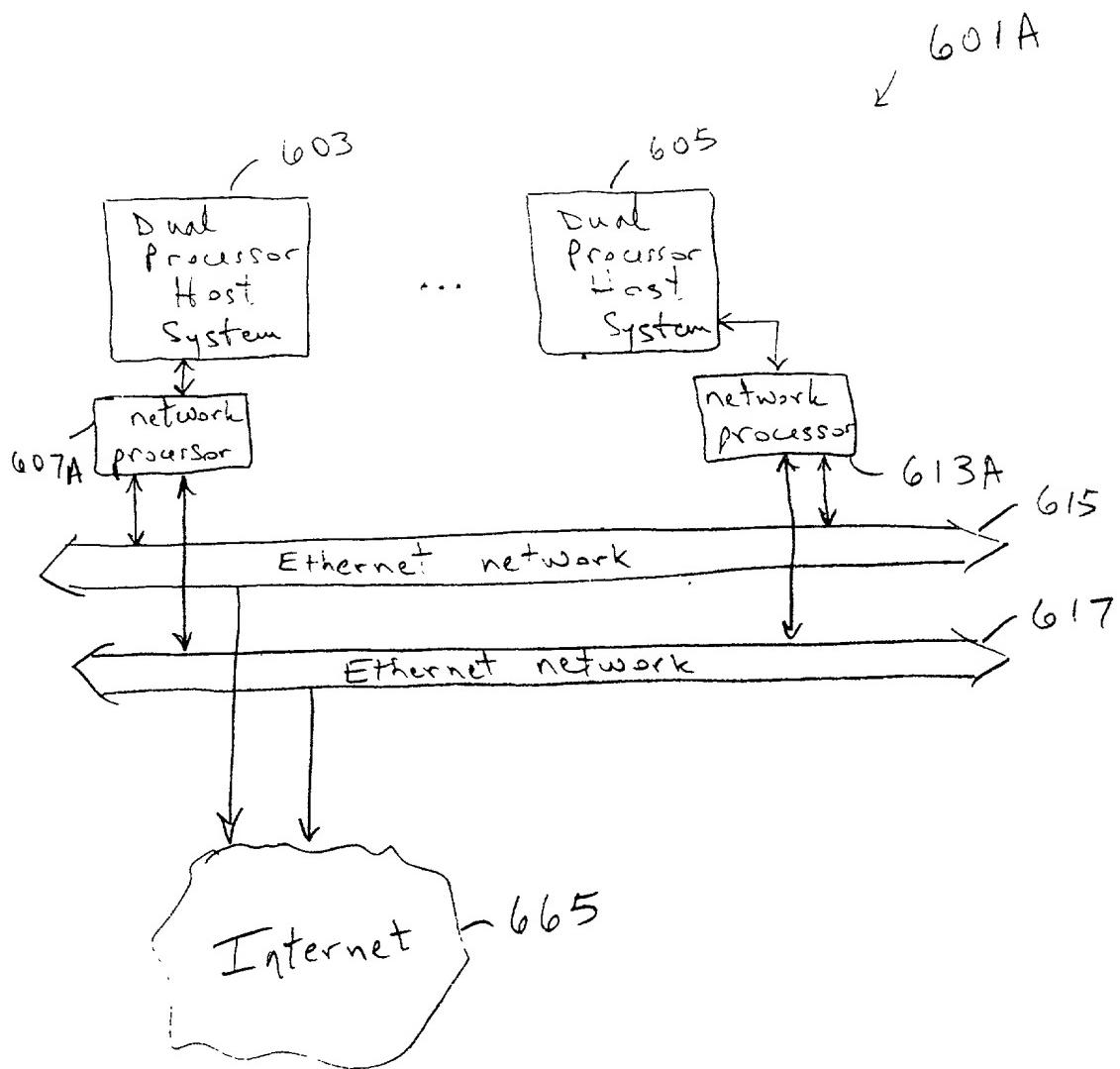


Fig. 17C

